REMARKS

Reconsideration and further examination of the application as amended are respectfully requested.

Claim Objections

Claims 15-17, 22-23, 28, 31, 34 and 37 were objected on the grounds of certain specified informalities. Applicant has amended claims 14-37 to correct the identified informalities. In particular, claims 14-37 were amended to correct the claim numbering, and claim 23 was amended to correct the preamble. Applicant submits that the claims as amended are in proper form and that the claim objection should be withdrawn.

§103

Claims 1-37 were rejected under 35 U.S.C. §103 as being obvious based on U.S. Patent No. 5,987,477 to Schmuck et al. (hereinafter "Schmuck"). For the following reasons applicant traverses the rejections.

Description of the Present Invention

The Applicant's invention provides for a system and method for a computerized file system. In one embodiment, a first process residing in a server node maintains a data file in that node. A second process generates a first message that requests the first process to grant to the second process a plurality of tokens that are required to modify at least one characteristic of the data file. In response to the request, the first process generates a second message that grants the tokens to the second process, if they are available. If any of the tokens are unavailable, i.e., one or more of the requested tokens are already granted

to a third process, the first process may generate a third message directing the third to process relinquish the unavailable tokens. In response, the third process may generate a fourth message relinquishing the current grant of these tokens.

In accordance with the invention, the number of tokens requested in the first message may be a plurality of tokens. Likewise, the third and fourth messages each may specify a plurality of tokens whose previous grant is to be revoked and relinquished, respectively.

Description of the Cited Reference

Schmuck teaches a shared disk file system where a file system instance on each machine has identical access to all of the disks coupled to and forming a part in the file system. The system allows parallel updates on the same file or directory in a shared disk environment.

The system provides a token manager which grants tokens to nodes. Nodes request a token for a specific mode. The token manager grants the token if the mode does not conflict with tokens granted to other nodes. If the requested token conflicts with a token granted to another node, a revoke is done and the conflicting node downgrades its token mode to a mode which does not conflict with the request mode.

Differences between the Present Invention and the Cited Reference

Claim 1 recites in relevant part as follows:

"a first process that maintains a data file in a computer-readable memory",

"a second process that generates **a first message** requesting that said second process be granted by said first process **a plurality of tokens** required for said second process to modify at least one characteristic of said file", and

"said first process generating **a second message** . . . that grants said **tokens** to said first process".

That is, the present invention positively recites a single message (the so-called first message) configured to carry a request for a plurality of tokens, and another single message (the so-called second message) configured to carry the plurality of requested tokens. The art of record, including Schmuck, fails to teach or suggest such the configuration of a single message configured to carry multiple tokens.

In particular, Schmuck, like the prior art referred to in the Background section of applicant's Specification, teaches a separate message for each token that is being requested by a given process. First of all, with Schmuck there is no server computer that stores the files to be accessed by one or more client computers. See Col. 5, lines 44-45 ("There is no file system server in the path for either data or metadata.") Instead, Schmuck discloses a parallel, shared disk file system in which each computer has direct access to the disks storing the files. See Col. 44, lines 59-62 and Fig. 1 ("In a parallel file system . . . all disks that make up the file system can independently be accessed by multiple processors"). Nevertheless, for each file stored in the set of disks, Schmuck discloses the election of a single computer, the "metadata node", which manages the metadata for the respective file and shares this information with other computers. See Col. 31, lines 25-28.

Schmuck also discloses the use of two different types of tokens, namely a "meta-node token" and a lock token. However, with Schmuck, a separate message is required to request and/or grant each of these types of tokens. In particular, a computer issues a first request message to a token manager (11) for a particular type of metadata token. The computer, seeking the metadata, can request only one of three different types of metadata tokens, a read-only (ro) token, a weak-write (ww) token or an exclusive-write (xw) token. See Col. 32, lines 11-26 (noting that a computer can either ask for the ww token or the xw token, but not both). In response, the token manager returns only a single token, either the "ro", the "ww" or "xw" token. See Col. 32, lines 13-18 (noting that the token manager, in response to a request for the "ww" token, grants either the "ww" token or the "ro" token), and Col. 32, lines 23-30 (noting that the token manager in response to a request for the "xw" token, grants either the "xw" token or the "ww" token). As shown, with Schmuck, each request and grant message directed to the token manager only carries a single metadata token.

Schmuck also discloses the need for a separate message directed to a different entity, the metanode itself as opposed to the token manager, to acquire lock tokens. Specifically, Schmuck describes five different types of lock tokens: a "rw" lock for read/write within the locally cached file size, a "rf" lock for reads beyond the locally cached file size, a "wf" lock for writes beyond the locally cached file size, a "wa" lock for writes that append a file, and a "xw" lock for writes that reduce the cached file size. See Col. 33, lines 4-14. A computer requesting one of these types of lock tokens issues a request, not to the token manager as is the case for metadata tokens, but instead to a lock

manager that is distributed across the computers. See Col. 32, line 67 to Col. 33, line 1 and Col. 34, line 1.

As shown, Schmuck fails to provides any teaching or suggestion for combining requests for its metanode token and its lock token into a single message. Indeed, because each of the requests are directed to different entities, i.e., the request for a metanode token goes to the token manager (11), while the request for a lock token goes to the lock manager, they must be carried and responded to with separate messages.

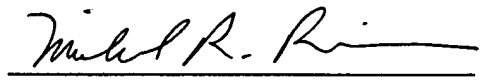
Furthermore, none of the portions of Schmuck identified in the Office Action at p. 4 support a teaching or suggestion of a single message configured to carry multiple tokens. In particular, Col. 31, lines 54-67, describes the three different types of metanode token, i.e., "ro", "ww", and "xw". Col. 32, lines 1-36 describes which metanode token is granted to a requesting computer. Col. 45, lines 3-11 and Col. 46, lines 55-59 are each directed to a discussion of quotas, and not tokens. As provided in Schmuck's Background section, quotas refer to limitations on how much data a particular user can access at any given time.

Because Schmuck fails to teach or suggest a single request message carrying a request for a plurality of tokens or a single response message carrying a grant of a plurality of tokens, it cannot render applicant's claimed invention obvious. Furthermore, because the other independent claims, i.e., claims 11, 13, 18-20, 26, 29, 32 and 35, also recite this feature, they too are allowable for the same reasons.

Applicant submits that the application as amended is in condition for allowance.

Please charge any additional fee occasioned by this paper to our Deposit Account No. 03-1237.

Respectfully submitted,



Michael R. Reinemann
Reg. No. 38,280
CESARI AND MCKENNA, LLP
88 Black Falcon Avenue
Boston, MA 02210-2414
(617) 951-2500

MARK-UP PAGES FOR THE APRIL 30, 2002, AMENDMENT TO U.S. PATENT APPLICATION SER. NO. 09/428,384

The replacement for claims 14-37 resulted from the following changes:

~~13~~14. A network computer system, comprising:

a first computer node having a data file in computer-readable memory; and

a second computer node that issues to the first computer node a first message requesting grant of a set of tokens required to carry out a modification of at least one characteristic of said file;

the first computer node issuing a second message to the second computer node after receipt of the first message, the second message granting the set of tokens to the first process if the set of tokens is available for grant to the second process.

~~14~~15. A system according to claim ~~13~~14, wherein:

the first computer node is a server node, and the second computer node is a non-server node.

~~15~~16. A system according to claim ~~13~~14, wherein:

the set of tokens comprises all tokens required to carry out the modification of the at least one characteristic of the file.

~~16~~17. A system according to claim ~~13~~14, wherein:

if at least one token in the set of tokens is unavailable for the grant because the at least one token is currently granted, the first computer node waits to issue the first message until after the first computer node receives a third message from a third computer node indicating relinquishment of current grant of the at least one token.

~~17~~18. A system according to claim ~~16~~17, wherein:

the at least one token comprises a plurality of tokens.

~~18~~19. Computer-readable memory containing computer-executable program instructions, the instructions comprising:

first instructions which when executed permit a data file to be maintained in computer storage memory;

second instructions which when executed generate a first message requesting grant of a plurality of tokens required to modify at least one characteristic of said file; and

third instructions which when executed generate a second message, in response to said first message, that grants said tokens if said tokens are available for grant to said second process.

~~19~~20. Computer-readable memory containing computer-executable program instructions, the instructions comprising:

first instructions which when executed generate a first message that grants a set of tokens, if the set of tokens is available for grant, to a requester of the set of tokens, the set of tokens being required to permit the requester to be able to modify at least one characteristic of a file stored in computer storage memory.

~~20~~21. Computer-readable memory containing computer-executable program instructions, the instructions comprising:

first instructions that when executed generate a request for grant of a set of tokens required to enable modification by an issuer of the request of at least one characteristic of a file residing in storage memory.

~~21~~22. Computer-readable memory according to Claim ~~18~~19, further comprising:

further instructions which when executed causes, if any of said tokens are unavailable for grant as a result of current grant of said tokens, generation of a third message revoking the current grant of said tokens.

~~22~~23. A ~~system~~ computer-readable memory according to claim ~~21~~22, wherein:

said further instructions, in response to said third message, generate a fourth message making said tokens available for grant.

~~23~~24. Computer-readable memory according to claim ~~19~~20, further comprising:

further instructions which when executed cause, if at least one token in the set of tokens is unavailable for grant because the at least one token is currently granted, generation of a second message that revokes previous grant of the at least one token prior to generating the first message.

~~24~~25. Computer-readable memory according to claim ~~19~~20, wherein:

the first message is generated in response to a request for the grant of the set of tokens generated, the request specifying all tokens required to be able to modify the at least one characteristic of the file.

~~25~~26. Computer-readable memory according to claim ~~20~~21, wherein:

the set of tokens comprises all tokens required to be able to modify the at least one characteristic of the file.

~~26~~27. A computerized data file system, comprising:

means for maintaining a data file in computer-readable memory; and

means for generating a first message requesting grant of a plurality of tokens required to modify at least on characteristic of said file;

means for generating a second message, in response to said first message, that grants said tokens if said tokens are available for grant.

~~27~~28. A system according to claim ~~26~~27, further comprising:

means for generating, if any of said tokens are unavailable for grant as a result of current grant of said tokens, a third message revoking the current grant of said tokens.

~~28~~29. A system according to claim ~~27~~28, further comprising:

means for generating, in response to said third message, a fourth message making said tokens available for grant.

~~29~~30. A computerized method for coherently maintaining and modifying a data file, comprising:

maintaining a data file in computer-readable memory;

generating a first message requesting grant of a plurality of tokens required to modify at least one characteristic of said file; and

generating a second message, in response to said first message, that grants said tokens if said tokens are available for grant.

~~30~~31. A method according to claim ~~29~~30, further comprising:

if any of said tokens are unavailable for grant as a result of current grant of said tokens to at least one other process, generating a third message revoking the grant of said tokens.

~~31~~32. A method according to claim ~~30~~31, wherein:

in response to said third message, a fourth message making said tokens available for grant is generated.

~~32~~33. A computerized method for use in maintaining coherency of a data file, comprising:

generating a first message that grants a set of tokens, if the set of tokens is available for grant, to a requester of the grant of the set of tokens, the set of tokens being required for requester to be able to modify at least one characteristic of the file.

~~33~~34. A method according to claim ~~32~~33, wherein:

if at least one token in the set of tokens is unavailable for grant because the at least one token has been currently granted, the method also comprises a second message that revokes current grant of the at least one token prior to generating the first message.

~~34~~35. A method according to claim ~~32~~33, wherein:

the first message is generated in response to a request for the grant of the set of tokens generated by the requester, the request specifying all tokens required for the requester to be able to modify the at least one characteristic of the file.

~~35~~36. A computerized method for use in maintaining coherency of a data file, comprising:

generating a request for grant of a set of tokens required to enable modification of at least one characteristic of the file.

~~36~~37. A method according to claim ~~35~~36, wherein:

the set of tokens comprises all tokens required to be able to modify the at least one characteristic of the file.